Accelerating Decentralized Deep Training with Sparse and Effective Topologies

Kun Yuan

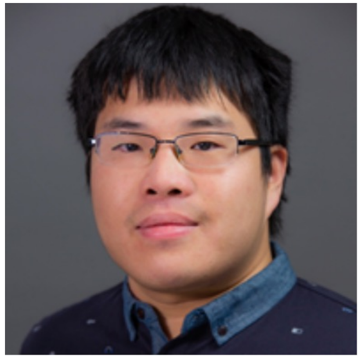
Center for Machine Learning Research @ Peking University

Nov. 23, 2022

Joint work with

Bicheng Ying (Google)

Yiming Chen (Alibaba)



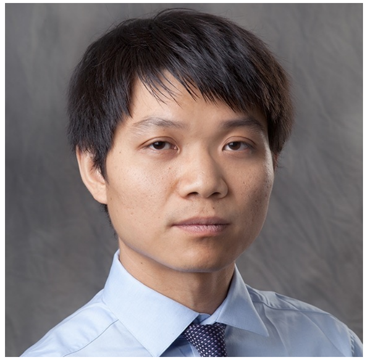
Hanbin Hu (Google)

Wotao Yin (Alibaba)

Zhuoqing Song (Fudan U)

Kexin Jin (Princeton U)

Weijian Li (Alibaba)

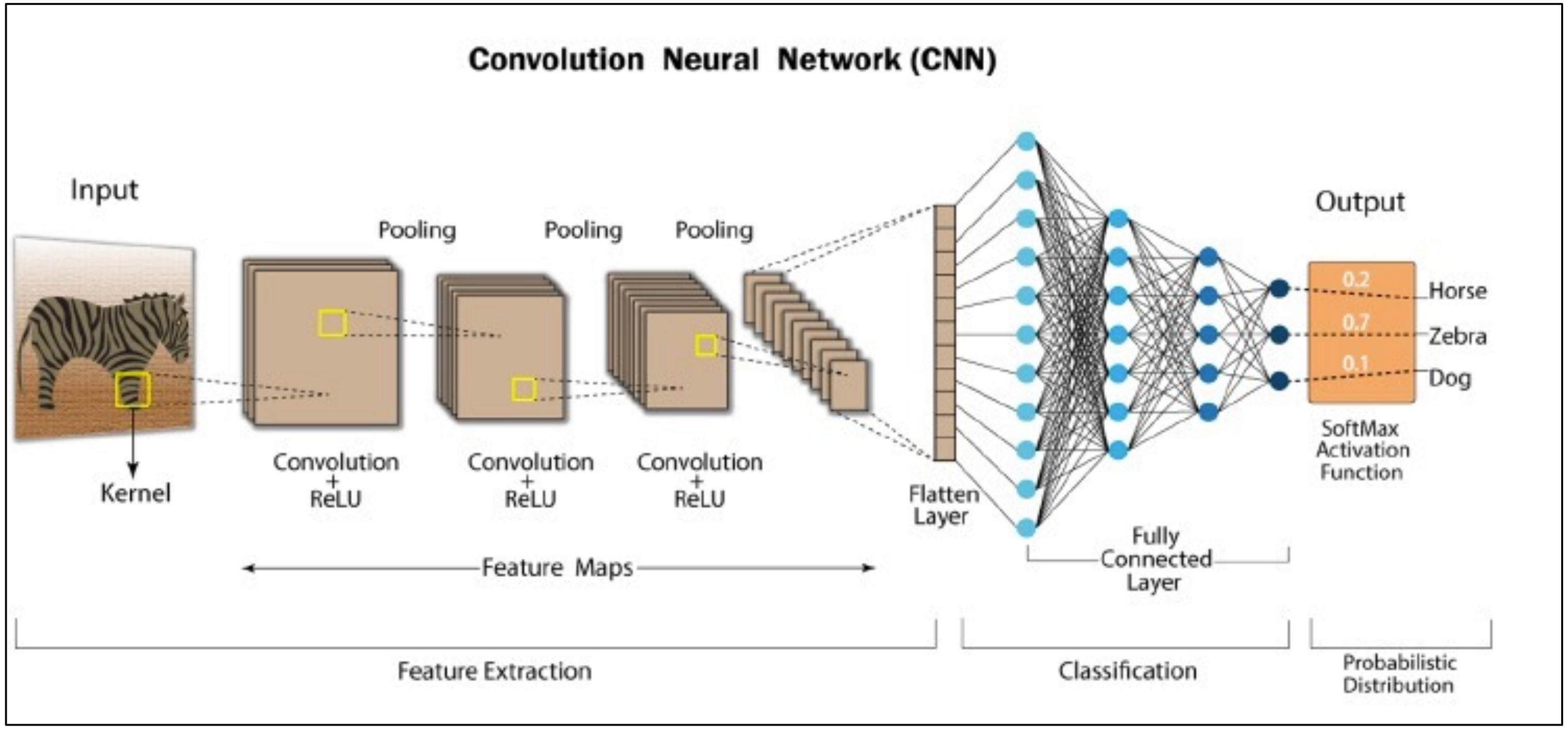


Ming Yan (CUHK-SZ)

Preface

Basics and Motivation

Training deep neural network is notoriously difficult



DNN training = non-convexity + **massive dataset** + huge models

Distributed learning

- Training deep neural networks typically requires **massive** datasets; efficient and scalable distributed optimization algorithms are in urgent need

- A network of n nodes (devices such as GPUs) collaborate to solve the problem:

$$\min_{x\in\mathbb{R}^d} \quad f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x), \quad \text{where} \quad f_i(x) = \mathbb{E}_{\xi_i\sim D_i} F(x;\xi_i).$$

  - Each component $f_i : \mathbb{R}^d \to \mathbb{R}$ is local and private to node i
  - Random variable ξ_i denotes the local data that follows distribution D_i
  - Each local distribution D_i is different; data heterogeneity exists

Vanilla parallel stochastic gradient descent (PSGD)

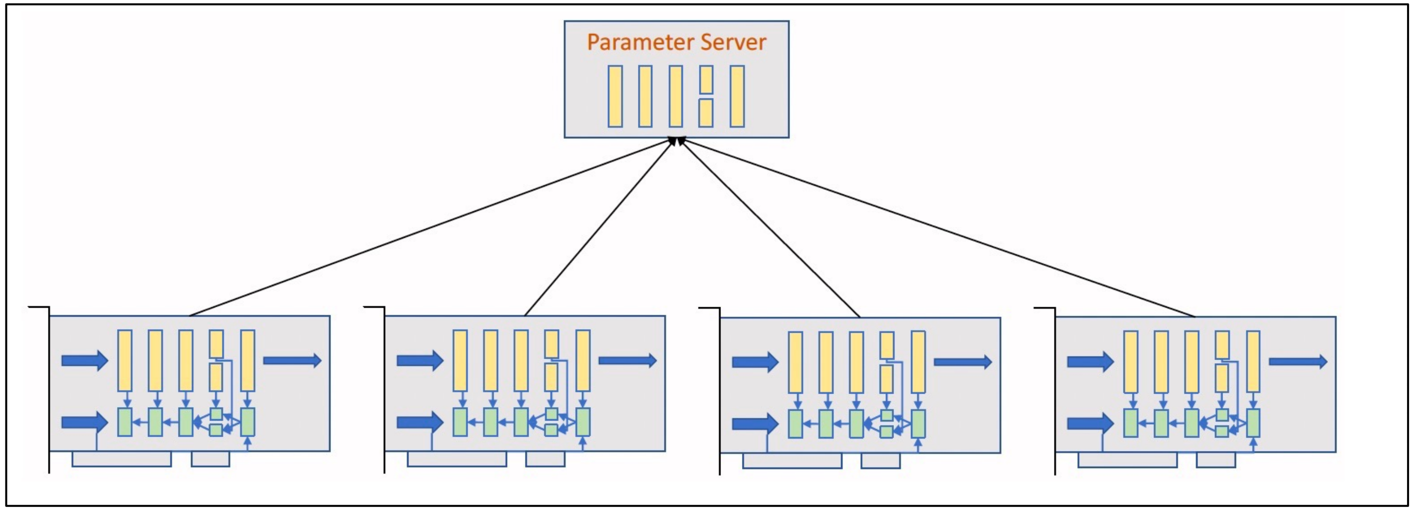
北京大学 PEKING UNIVERSITY

$$g_i^{(k)} = \nabla F(x^{(k)}; \xi_i^{(k)}) \qquad \text{(Local compt.)}$$

$$x^{(k+1)} = x^{(k)} - \frac{\gamma}{n} \sum_{i=1}^{n} g_i^{(k)} \qquad \text{(Global comm.)}$$

- Each node i samples data $\xi_i^{(k)}$ and computes gradient $\nabla F(x^{(k)}; \xi_i^{(k)})$
- All nodes synchronize (i.e. globally average) to update model x per iteration

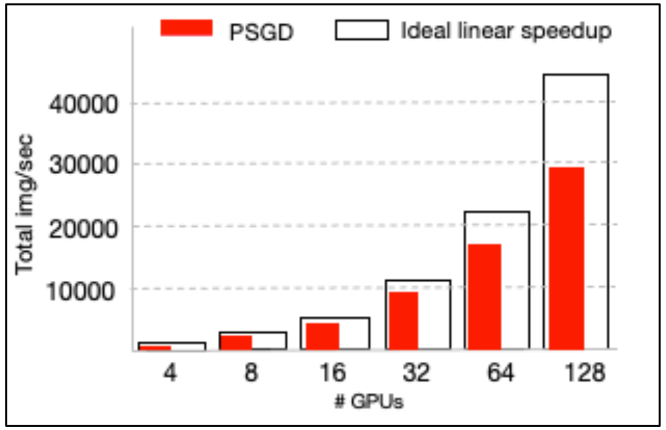
Vanilla parallel stochastic gradient descent (PSGD)



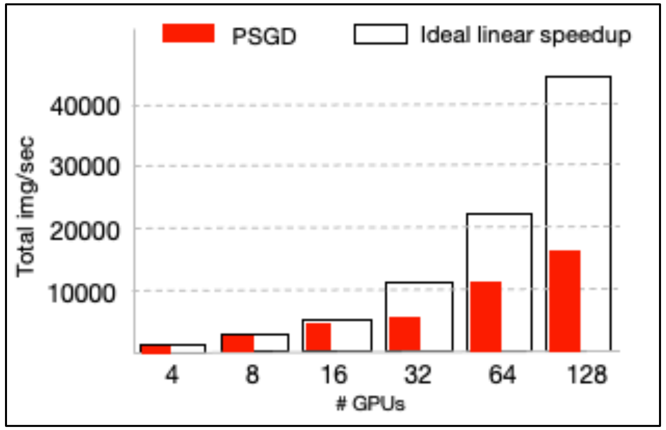
- Global average incurs $O(n)$ comm. overhead; **proportional to network size n**
- When network size n is large, PSGD suffers severe communication overhead

PSGD cannot achieve linear speedup due to comm. overhead

- PSGD cannot achieve ideal linear speedup in throughput due to comm. overhead
- Larger comm-to-compt ratio leads to worse performance in PSGD



Small comm.-to-compt. ratio



Large comm.-to-compt. ratio

- How can we accelerate PSGD? **We must reduce communication overhead.**

B. Ying, K. Yuan, H. Hu, Y. Chen and W. Yin, "BlueFog: Make decentralized algorithms practical for optimization and deep learning", arXiv: 2111. 04287, 2021

Methodologies to save communication

- Each node sends a full model (or gradient) to the server; proportional to dimension d

  [Communication compression]

- Each node interacts with the server at every iteration; proportional to iteration numbers

  [Lazy communication]

- Global average incurs $O(n)$ comm. overhead; proportional to network size n

  [Decentralized communication]

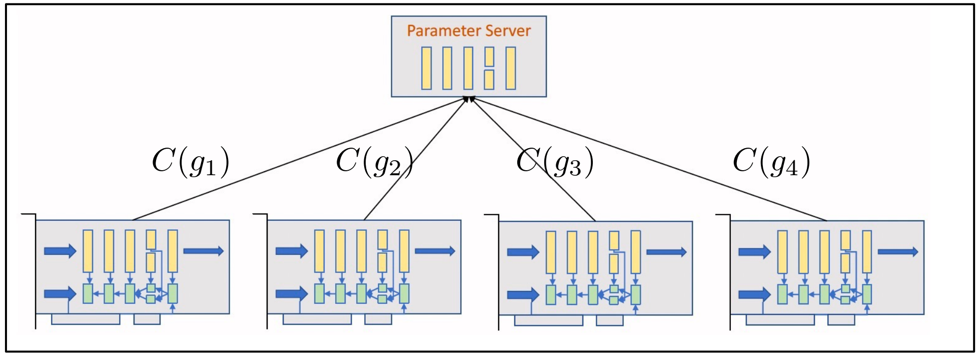
Communication compression

PEKING UNIVERSITY 北京大学

- A basic (but not state-of-the-art) algorithm is QSGD [Alistarh et. al., 2017]

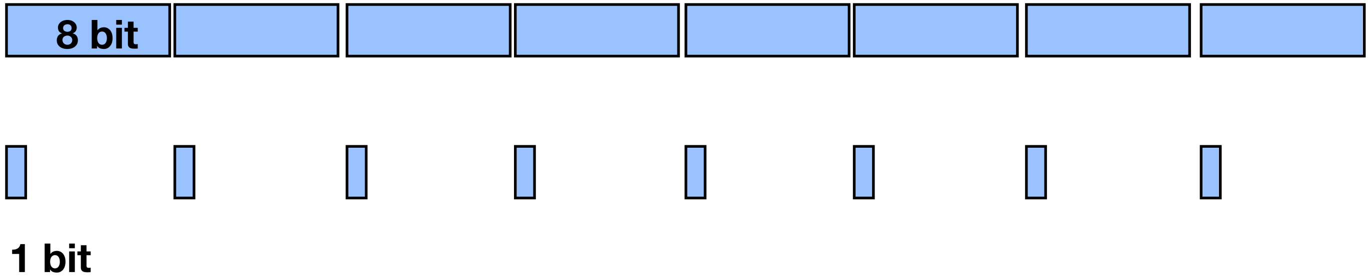
$$g_i^{(k)} = \nabla F(x_i^{(k)}; \xi_i^{(k)})$$

$$x_i^{(k+1)} = x_i^{(k)} - \frac{\gamma}{n}\sum_{j=1}^{n} C(g_j^{(k)})$$



- $C(\cdot)$ is a compressor. It can quantize or sparsify the full gradient

Quantization

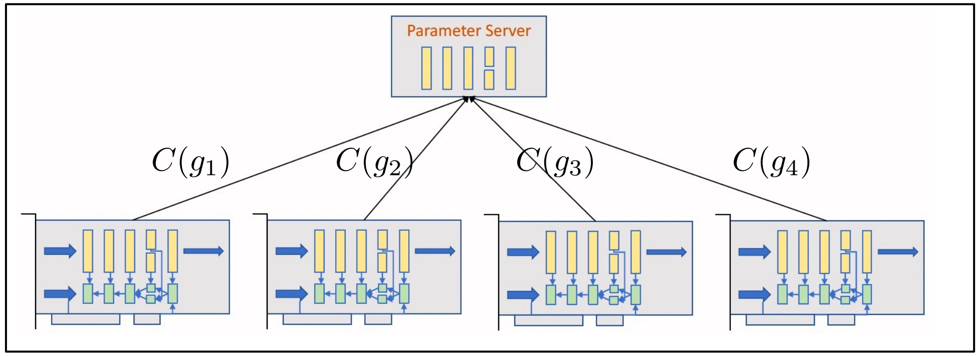


Communication compression

- A basic (but not state-of-the-art) algorithm is QSGD [Alistarh et. al., 2017]

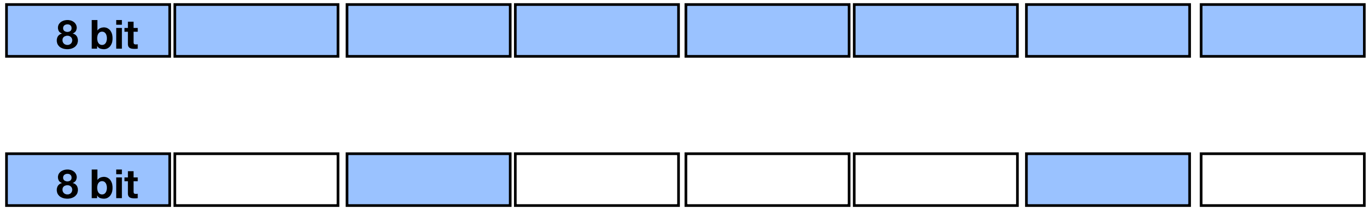
$$g_i^{(k)} = \nabla F(x_i^{(k)}; \xi_i^{(k)})$$

$$x_i^{(k+1)} = x_i^{(k)} - \frac{\gamma}{n}\sum_{j=1}^{n} C(g_j^{(k)})$$



- $C(\cdot)$ is a compressor. It can quantize or sparsify the full gradient

Sparsification



Lazy communication (Federated Average)

$$x_i^{(k+\frac{1}{2})} = x_i^{(k)} - \gamma \nabla F(x_i^{(k)}; \xi_i^{(k)}) \quad \text{(Local update)}$$

$$x_i^{(k+1)} = \begin{cases} x_i^{(k+\frac{1}{2})} & \text{if } \mathrm{mod}(k,\tau) \neq 0 \\ \frac{1}{n}\sum_{j=1}^{n} x_j^{(k+\frac{1}{2})} & \text{if } \mathrm{mod}(k,\tau) = 0 \end{cases} \quad \text{(Lazy comm.)}$$

- Nodes communicate once every τ iterations [Konecny et .al. 2015, 2016]
- Or nodes communicate when necessary, i.e., [Chen et. al. 2018; Liu et.al., 2019]
- In ProxSkip [Mishchenko et. al., 2022], lazy strategy is proved to save communication

北京大學 PEKING UNIVERSITY

This talk will study distributed learning with **decentralized communication**

Contents

- Decentralized SGD and topology effects
- Exponential graphs are provably efficient
- EquiTopo graphs are new state-of-the-art
- BlueFog libraries: introduction and demos

PART 01

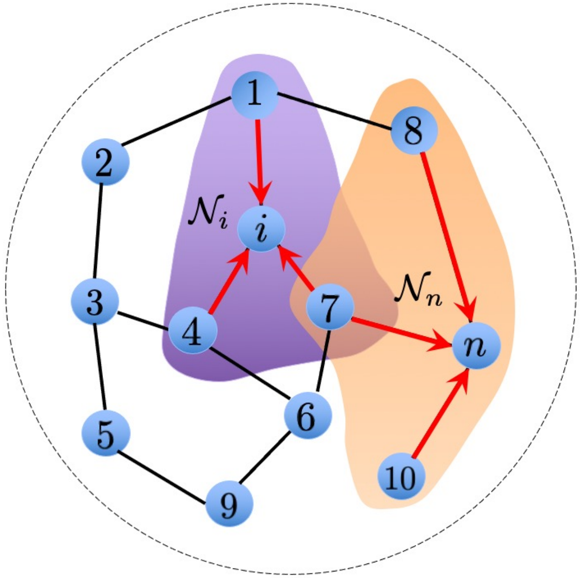
Decentralized SGD and Topology Effects

Decentralized SGD (DSGD)

- To break $O(n)$ comm. overhead, we replace global average with partial average

$$x_i^{(k+\frac{1}{2})} = x_i^{(k)} - \gamma \nabla F(x_i^{(k)}; \xi_i^{(k)}) \quad \text{(Local update)}$$

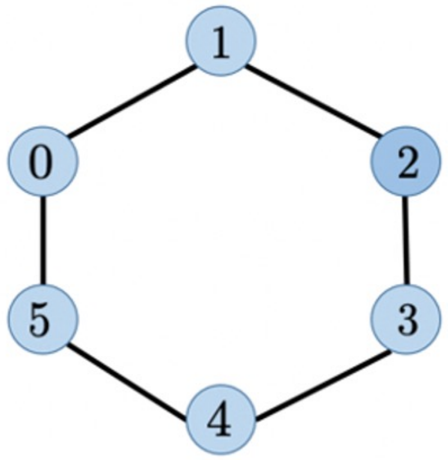
$$x_i^{(k+1)} = \sum_{j \in \mathcal{N}_i} w_{ij} x_j^{(k+\frac{1}{2})} \quad \text{(Partial averaging)}$$



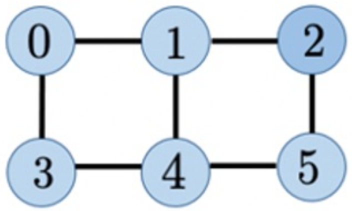
- DSGD = local SGD update + partial averaging [LS08]
- $\mathcal{N}_i$ is the set of neighbors at node i ; w_{ij} scales information from j to i
- Incurs $O(d_{\max})$ comm. overhead per iteration where $d_{\max} = \max_i\{|\mathcal{N}_i|\}$ is the graph maximum degree

DSGD is more communication-efficient than PSGD

- Incurs $O(1)$ comm. overhead on **sparse** topologies; much less than global average $O(n)$



Ring



Grid

DSGD is more communication-efficient than PSGD

- A real experiment on a 256-GPUs cluster [CYZ+21]

Model	Ring-Allreduce	Partial average
ResNet-50 (25.5M)	278 ms	150 ms
Bert (300M)	1469 ms	567 ms

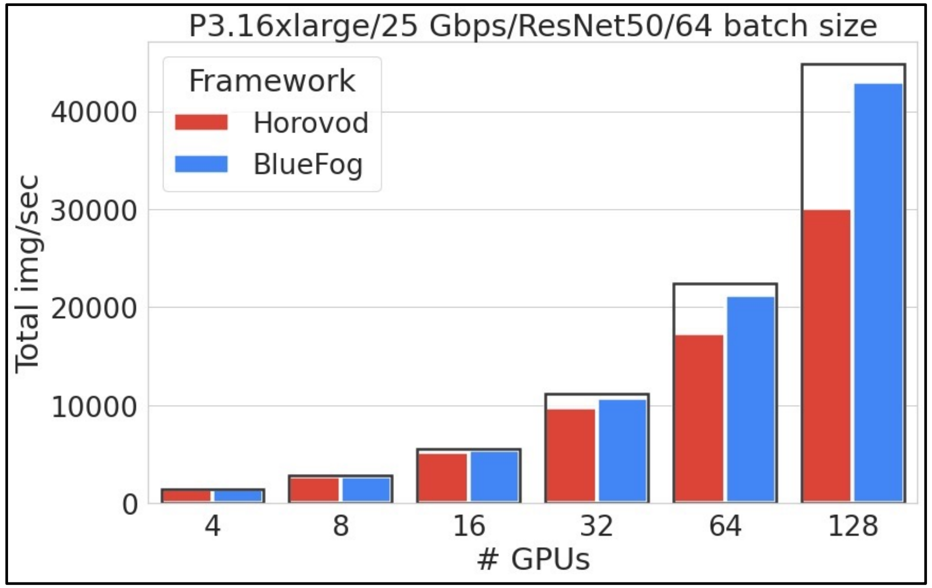
Table. Comparison of per-iter comm. time in terms of runtime with 256 GPUs

- DSGD saves more communications per iteration for larger models

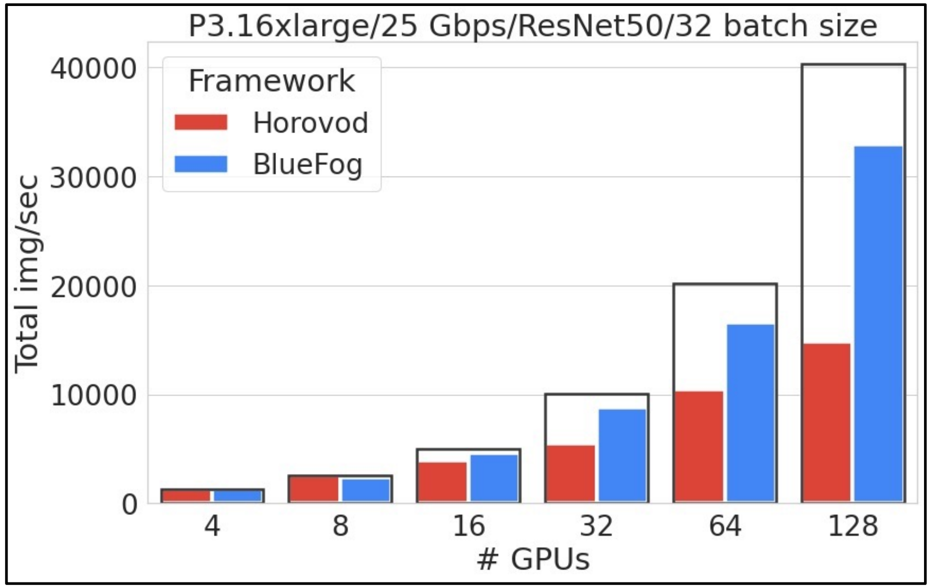
[CYZ+21] Y. Chen*, K. Yuan*, Y. Zhang, P. Pan, Y. Xu, and W. Yin, ``Accelerating Gossip SGD with Periodic Global Averaging”, ICML 2021

DSGD is more communication-efficient than PSGD

- DSGD (BlueFog) has **better linear speedup** than PSGD (Horovod) due to its small comm. overhead



Small comm.-to-compt. ratio



Large comm.-to-compt. ratio

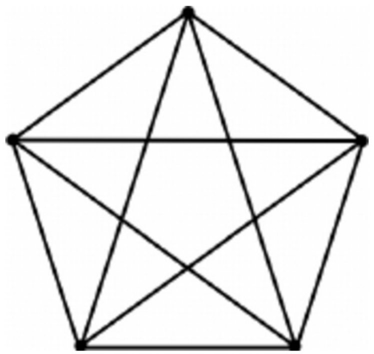
B. Ying, K. Yuan, H. Hu, Y. Chen and W. Yin, "BlueFog: Make decentralized algorithms practical for optimization and deep learning", arXiv: 2111. 04287, 2021

However, DSGD has slower convergence

- The efficient comm. comes with a cost: slower convergence
- Partial average $x_i^+ = \sum w_{ij} x_j$ is less effective to aggregate information than global average
- The average effectiveness can be evaluated by **graph spectral gap**:

$$\rho = \|W - \frac{1}{n}\mathbb{1}\mathbb{1}^T\|_2 \in (0,1) \ \text{ where } W = [w_{ij}] \in \mathbb{R}^{n \times n}$$

Fully-connected matrix



$$W = \frac{1}{5}\mathbb{1}\mathbb{1}^T = \begin{bmatrix} \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \end{bmatrix}$$

However, DSGD has slower convergence

- The efficient comm. comes with a cost: slower convergence
- Partial average $x_i^+ = \sum w_{ij} x_j$ is less effective to aggregate information than global average
- The average effectiveness can be evaluated by **graph spectral gap**:

$$\rho = \|W - \frac{1}{n}\mathbb{1}\mathbb{1}^T\|_2 \in (0, 1) \ \text{ where } W = [w_{ij}] \in \mathbb{R}^{n \times n}$$

  - Well-connected topology has $\rho \to 0$, e.g. fully-connected topology
  - Sparsely-connected topology has $\rho \to 1$, e.g. ring has $\rho = O(1 - \frac{1}{n^2})$
  - ρ or $1 - \rho$ essentially gauges the **graph connectivity**

DSGD convergence rate

- Convergence comparison (non-convex and data-homogeneous scenario) [KLB+20]:

$$\text{P-SGD}: \quad \frac{1}{T}\sum_{k=1}^{T}\mathbb{E}\|\nabla f(\bar{x}^{(k)})\|^2 = O\Big(\frac{\sigma}{\sqrt{nT}}\Big)$$

$$\text{D-SGD}: \quad \frac{1}{T}\sum_{k=1}^{T}\mathbb{E}\|\nabla f(\bar{x}^{(k)})\|^2 = O\Big(\frac{\sigma}{\sqrt{nT}} + \underbrace{\frac{\rho^{2/3}\sigma^{2/3}}{T^{2/3}(1-\rho)^{1/3}}}_{\text{extra overhead}}\Big)$$

where σ^2 is the gradient noise, and T is the number of iterations

- D-SGD can asymptotically converge as fast as P-SGD when $T \to \infty$; the first term dominates; reach **linear speedup** asymptotically

- But D-SGD **requires more iteration** to reach that stage due to the overhead caused by partial average

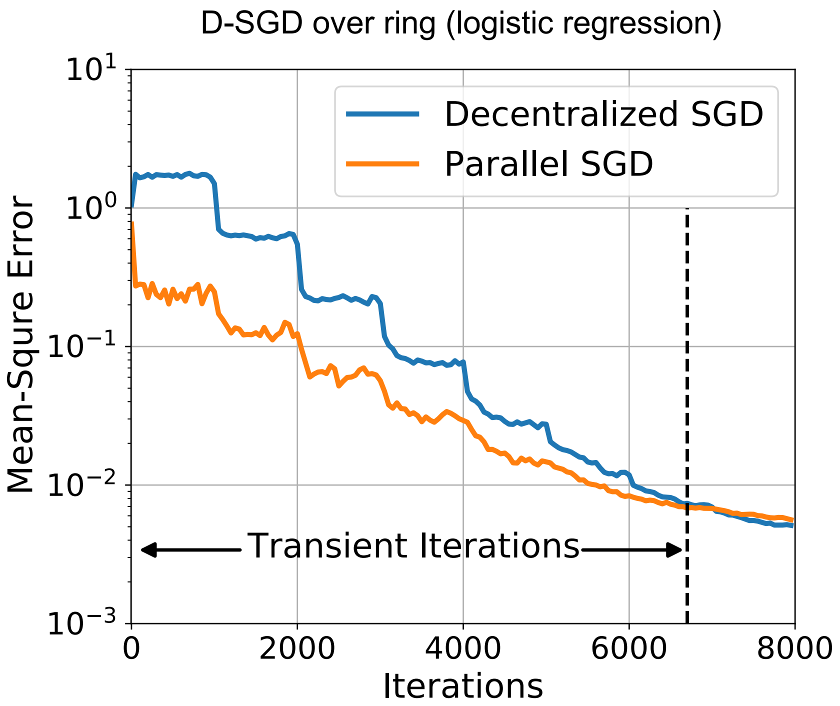
Transient iterations

- Definition [POP21]: number of iterations before D-SGD achieves linear speedup

- D-SGD for non-convex and data-homogeneous scenario has $O(n^3(1-\rho)^{-2})$ transient iterations

$$\frac{\rho^{2/3}\sigma^{2/3}}{T^{2/3}(1-\rho)^{1/3}} \le \frac{\sigma}{\sqrt{nT}} \quad \Longrightarrow \quad O(\frac{\rho^4 n^3}{(1-\rho)^2})$$

- Sparse topology $\rho \to 1$ incurs longer tran. Iters.



Trade-off between comm. cost and trans. iters.

PEKING UNIVERSITY

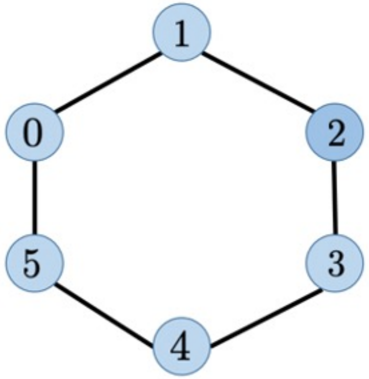
- Recall per-iter comm. $O(d_{\max})$ and trans. iters. $\Omega(n^3(1-\rho)^{-2})$

- Trade-off between per-iteration communication and transient iteration complexity

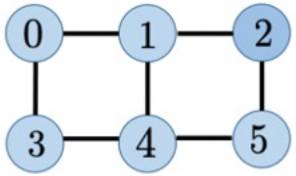
	Sparse topology	Dense topology
per-iter comm.	$\checkmark$	$\times$
trans. iter. complexity	$\times$	$\checkmark$

What topology to use?

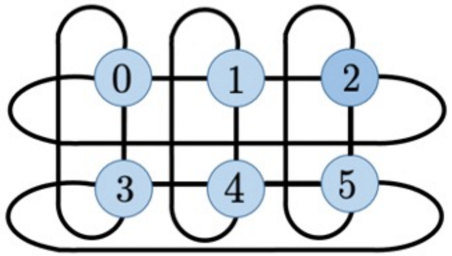
- Shall we use these common topologies to organize all nodes?



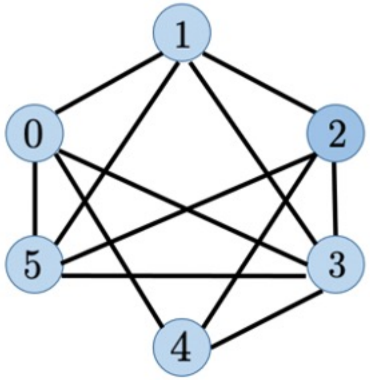
Ring



Grid



Torus



Erdos-Renyi Random

What topology to use?

- Communication cost v.s. transient iteration complexity in DSGD

Topology	Per-iter. Comm.	Trans. Iters. (iid scenario)
Ring	$O(1)$	$O(n^7)$
2D-Grid	$O(1)$	$\tilde{O}(n^5)$
2D-Torus	$O(1)$	$O(n^5)$
$\frac{1}{2}$-RandGraph	$O(n)$	$O(n^3)$

The smaller both comm. cost and tran. Iters. are, the better

- These topologies either have expensive communication cost or longer transient stage

- **Is there any topology that enables both cheap communication and fast convergence?**

PART 02

Exponential graphs are provably efficient

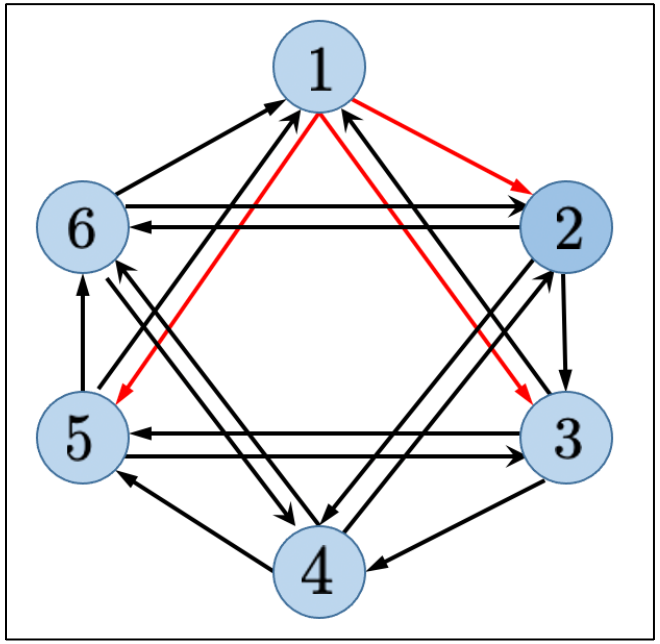
Static exponential graph: topology and per-iteration comm.

- Each node links to neighbors that are $2^0, 2^1, \cdots, 2^{\lfloor \log_2(n-1) \rfloor}$ away [ALB+19]

- In the figure, node 1 connects to node 2, 3 and 5.

- Each node has $\lceil \log_2(n) \rceil$ neighbors; per-iter comm. cost is $O(\log_2(n))$

- Empirically successful in deep training but less theoretically understood

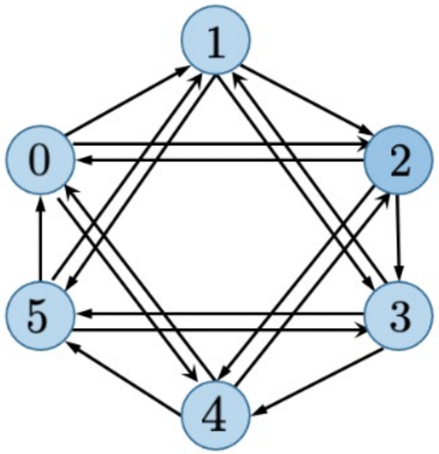


Static exponential graph: weight matrix

- The weight matrix associated with exponential graph is defined as

$$w_{ij}^{\exp} = \begin{cases} \frac{1}{\lceil \log_2(n) \rceil + 1} & \text{if } \log_2(\mathrm{mod}(j-i,n)) \text{ is an integer or } i=j \\ 0 & \text{otherwise.} \end{cases}$$

- An illustrating example:



$$W = \begin{bmatrix} \frac{1}{4} & 0 & \frac{1}{4} & 0 & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} \\ \frac{1}{4} & 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 \\ 0 & \frac{1}{4} & 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}$$

Static exponential graph: connectivity

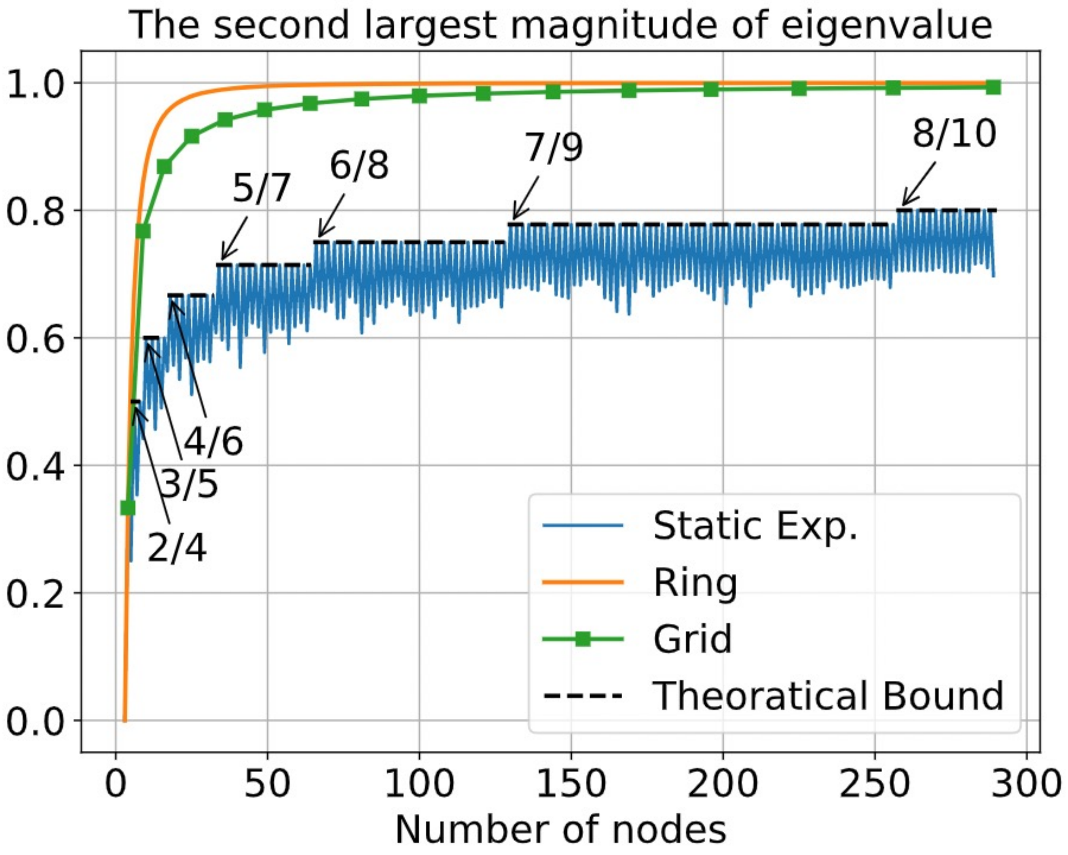
- Is the static exponential graph well connected?

Theorem. Let $\tau = \lceil \log_2(n) \rceil$, and $\rho = \|W - \frac{1}{n}\mathbb{1}\mathbb{1}^T\|_2$ be the spectral gap. It holds that

$$\begin{cases} \rho = 1 - \dfrac{2}{\tau+1}, & \text{when } n \text{ is even} \\ \rho < 1 - \dfrac{2}{\tau+1}, & \text{when } n \text{ is odd} \end{cases}$$

- This theorem implies that exponential graph has $\rho(W) = O(1 - 1/\log_2(n))$
- Highly non-trivial proofs; requires smart utilization of Fourier transform

Static exponential graph: illustration of the spectral gap



- Our theoretical bound is very tight
- Spectral gap increases slowly when n grows

Static exponential graph: transient iterations in DSGD

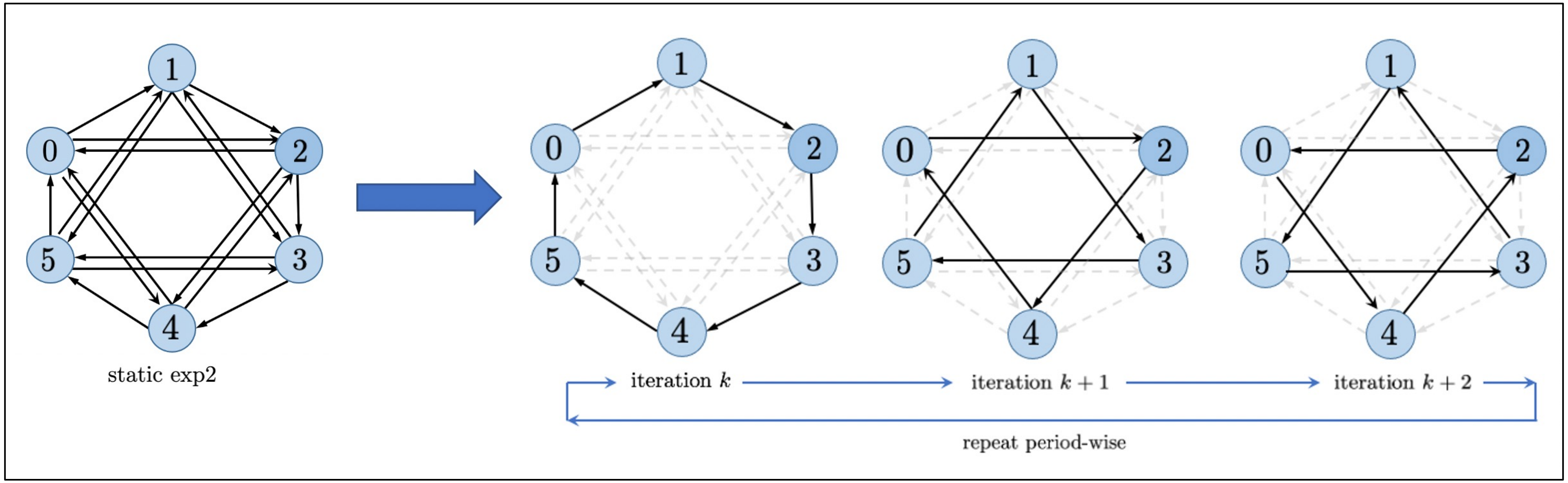
- Recall DSGD has transient iteration complexity $O(n^3/(1-\rho)^2)$ in iid scenarios
- With $\rho(W) = O(1 - 1/\log_2(n))$, exponential graphs have tran. iters. as $O(n^3 \log_2^2(n))$
- Per-iteration communication and transient iteration complexity are **nearly the best** (up to $\log_2(n)$)

Topology	Per-iter. Comm.	Trans. Iters. (iid scenario)
Ring	$O(1)$	$O(n^7)$
2D-Grid	$O(1)$	$\tilde{O}(n^5)$
2D-Torus	$O(1)$	$O(n^5)$
$\frac{1}{2}$-RandGraph	$O(n)$	$O(n^3)$
Static Exp	$\tilde{O}(1)$	$\tilde{O}(n^3)$

- **Can we achieve even better topology?**

One-peer exponential graph: topology

- **Split** exponential graph into a sequence of one-peer realizations;
- Each node has **exactly one** neighbor per iteration
- **O(1) per-iteration communication**; same as ring; cheaper than grid

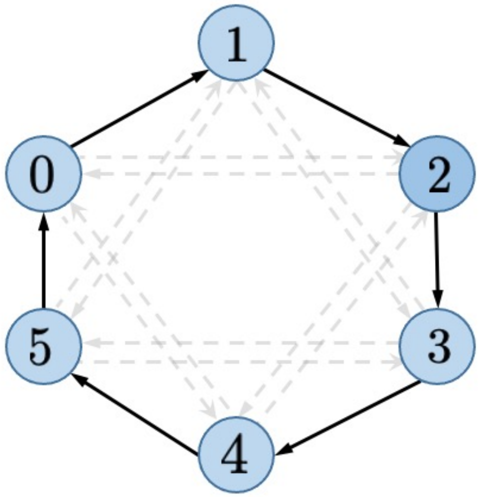


One-peer exponential graph: weight matrix

- We let $\tau = \lceil \log_2(n) \rceil$. The weight matrix $W^{(k)}$ is defined as

$$w_{ij}^{(k)} = \begin{cases} \frac{1}{2} & \text{if } \log_2(\text{mod}(j-i,n)) = \text{mod}(k,\tau) \\ \frac{1}{2} & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

- An illustrating example



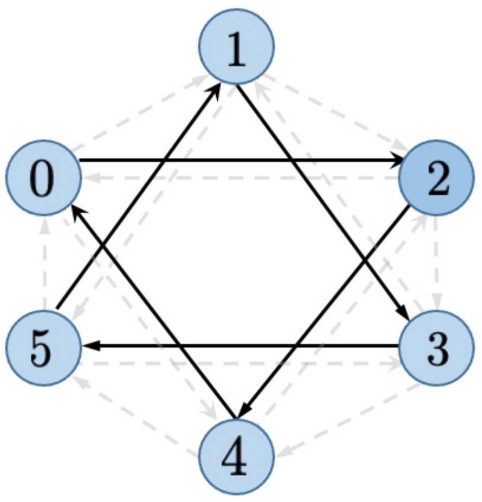
$$W = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

One-peer exponential graph: weight matrix

- We let $\tau = \lceil \log_2(n) \rceil$. The weight matrix $W^{(k)}$ is defined as

$$w_{ij}^{(k)} = \begin{cases} \frac{1}{2} & \text{if } \log_2(\mathrm{mod}(j-i, n)) = \mathrm{mod}(k, \tau) \\ \frac{1}{2} & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

- An illustrating example



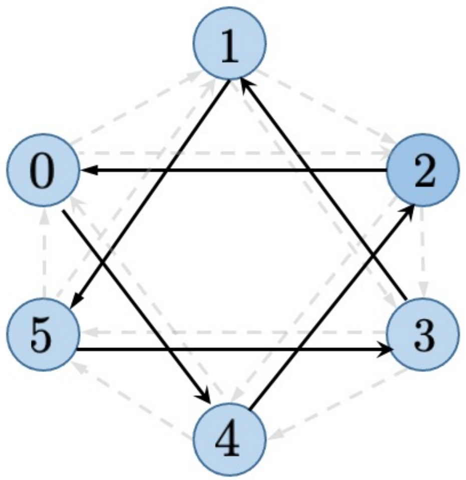
$$W = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix}$$

One-peer exponential graph: weight matrix

- We let $\tau = \lceil \log_2(n) \rceil$. The weight matrix $W^{(k)}$ is defined as

$$
w_{ij}^{(k)} = \begin{cases} \frac{1}{2} & \text{if } \log_2(\mathrm{mod}(j-i,n)) = \mathrm{mod}(k,\tau) \\ \frac{1}{2} & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}
$$

- An illustrating example



$$
W = \begin{bmatrix} \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}
$$

DSGD over one-peer exponential graph

北京大学 PEKING UNIVERSITY

Sample $W^{(k)}$ over one-peer exponential graph

$$x_i^{(k+\frac{1}{2})} = x_i^{(k)} - \gamma \nabla F(x_i^{(k)}; \xi_i^{(k)}) \quad \text{(Local update)}$$

$$x_i^{(k+1)} = \sum_{j \in \mathcal{N}_i} w_{ij}^{(k)} x_j^{(k+\frac{1}{2})} \quad \text{(Partial averaging)}$$

- DSGD with **time-varying** weight matrix;
- Per-iteration communication cost is **O(1)**; very efficient

One-peer exponential graph: Periodic exact average

- While one-peer exponential graph is **sparse**, it is **effective** to aggregate information

Theorem. Suppose $\tau = \log_2(n)$ is a positive integer. It holds that

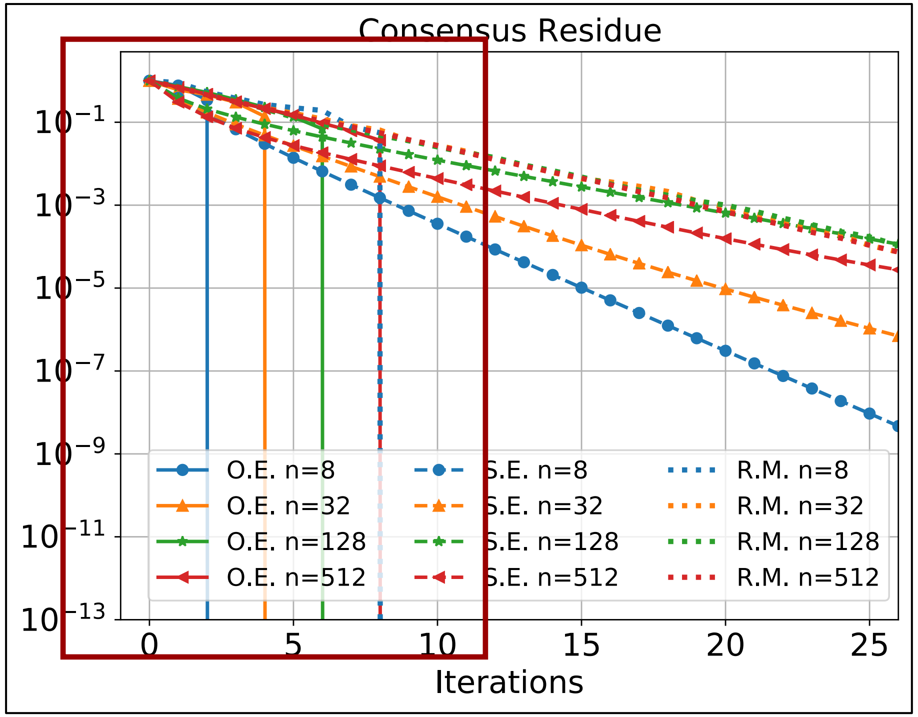
$$W^{(k+\ell)}W^{(k+\ell-1)}\cdots W^{(k+1)}W^{(k)} = \frac{1}{n}\mathbb{1}\mathbb{1}^T$$

for any integer $k \geq 0$ and $\ell \geq \tau - 1$.

- While each realization is sparser, a sequence (with length τ) of one-peer graphs will enable effective global averaging

One-peer exponential graph: Periodic exact average

- We examine $\|\frac{1}{n}\mathbb{1}\mathbb{1}^T x - \prod_{k=0}^{T} W^{(k)} x\|$ for a vector x
- $\frac{1}{n}\mathbb{1}\mathbb{1}^T x$ is the global average
- $\prod_{k=0}^{T} W^{(k)} x$ is the partial average after T iterations
- One-peer exp. achieves global average after $\log_2(n)$ iters.



Apply one-peer exponential graph to DSGD

Assumption *(1) Each $f_i(x)$ is L-smooth; (2) Each gradient noise is unbiased and has bounded variance σ^2; (3) Each local distribution D_i is identical (iid)*

Theorem *Under the above assumptions and with $\gamma = O(1/\sqrt{T})$, let $\tau = \log_2(n)$ be an integer, DSGD with one-peer exponential graph will converge at*

$$\frac{1}{T}\sum_{k=1}^{T}\mathbb{E}\|\nabla f(\bar{x}^{(k)})\|^2 = O\Big(\frac{\sigma}{\sqrt{nT}} + \underbrace{\frac{\sigma^{2/3}\log_2^{1/3}(n)}{T^{2/3}}}_{\text{extra overhead}}\Big)$$

Novel analysis; require new tricks to utilize periodic exact average to establish tight convergence

Static v.s. one-peer exponential graph

- Convergence rate of DSGD over static and one-peer exponential graphs are

$$\text{Static exp.} \quad O\Big(\frac{\sigma}{\sqrt{nT}} + \frac{\sigma^{2/3}}{T^{2/3}(1-\rho)^{1/3}}\Big) \quad \text{(where } 1-\rho = O(1/\log_2(n))\text{)}$$

$$\text{One-peer exp.} \quad O\Big(\frac{\sigma}{\sqrt{nT}} + \frac{\sigma^{2/3}\log_2^{1/3}(n)}{T^{2/3}}\Big)$$

- DSGD with one-peer exp. converges **as fast as** static exp.; **a surprising result**.

- DSGD with both graphs are with the **same** transient iteration complexity $O(n^3\log_2^2(n))$

- The communication cost saving in one-peer exponential graph is a **free lunch**

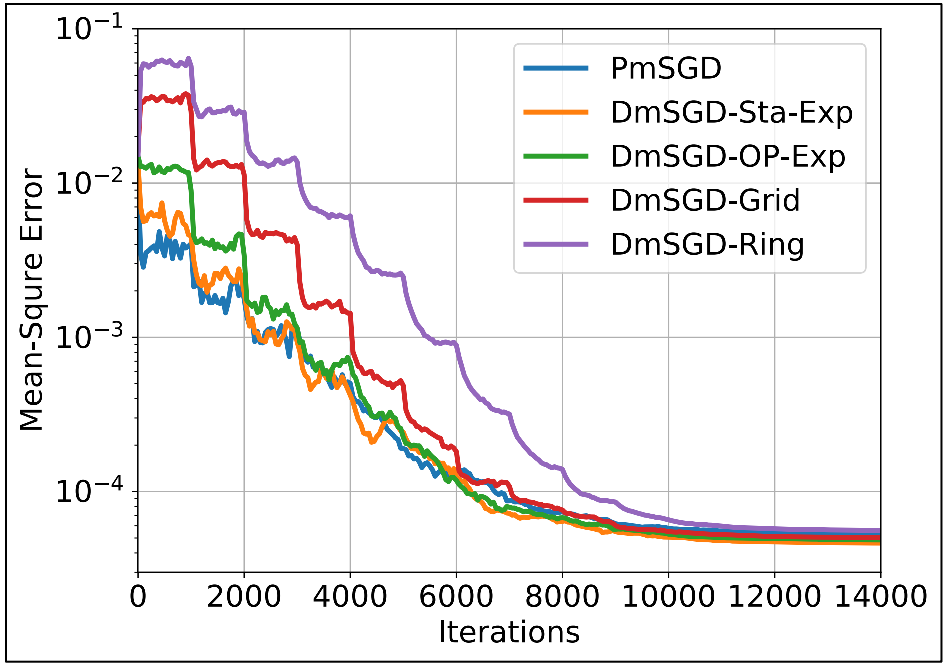
Compare one-peer exp. with other topologies

Topology	Per-iter. Comm.	Trans. Iters. (iid scenario)
Ring	$O(1)$	$O(n^7)$
2D-Grid	$O(1)$	$\tilde{O}(n^5)$
2D-Torus	$O(1)$	$O(n^5)$
$\frac{1}{2}$-RandGraph	$O(n)$	$O(n^3)$
Static Exp	$\tilde{O}(1)$	$\tilde{O}(n^3)$
One-Peer Expo	$O(1)$	$\tilde{O}(n^3)$

We recommend using one-peer exponential graph in deep training.

Exponential graphs have shorter tran. iters.

- Illustration of the tran. iters. on DSGD (momentum version) for logistic regression
- DSGD over one-peer exponential graph converges faster than other topologies



Comparison over 32 nodes

Experiments in deep training (image classification)

ImageNet-1K dataset

1.3M training images

50K test images

1K classes

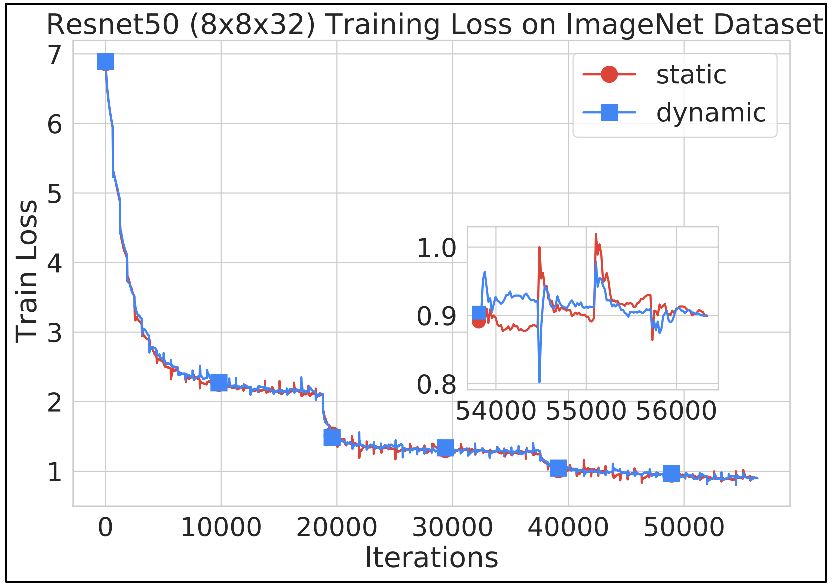
DNN model: ResNet-50 (25.5M parameters)

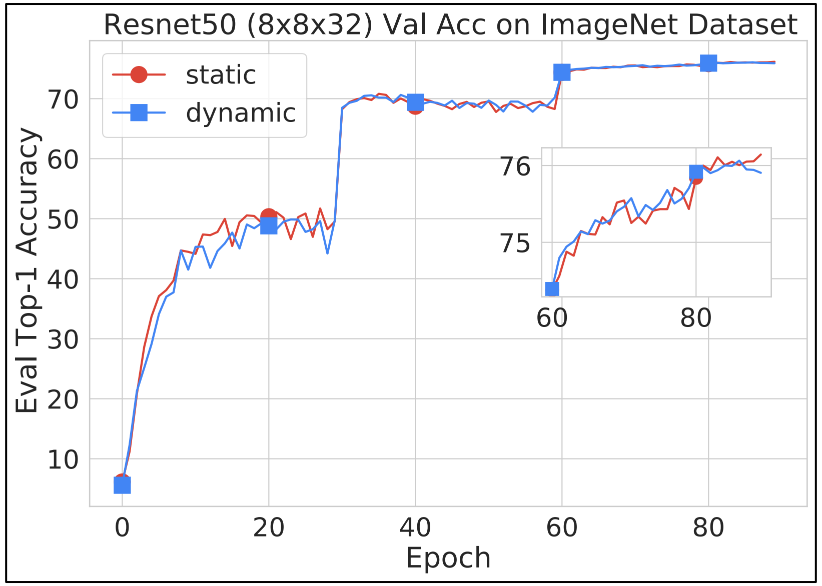
GPU: Up to 256 Tesla V100 GPUs

- **Wall-clock time** to finish 90 epochs of training; measures per-iter communication
- **Validation accuracy** after 90 epochs of training; measures convergence rate

One peer is not slower than static exponential graph

Image classification: ResNet-50 for ImageNet; $8 \times 8 = 64$ GPUs.





One-peer and exponential graphs converge **roughly the same**; but one-peer is more comm. efficient

DSGD over one-peer Exp. achieves better linear speedup

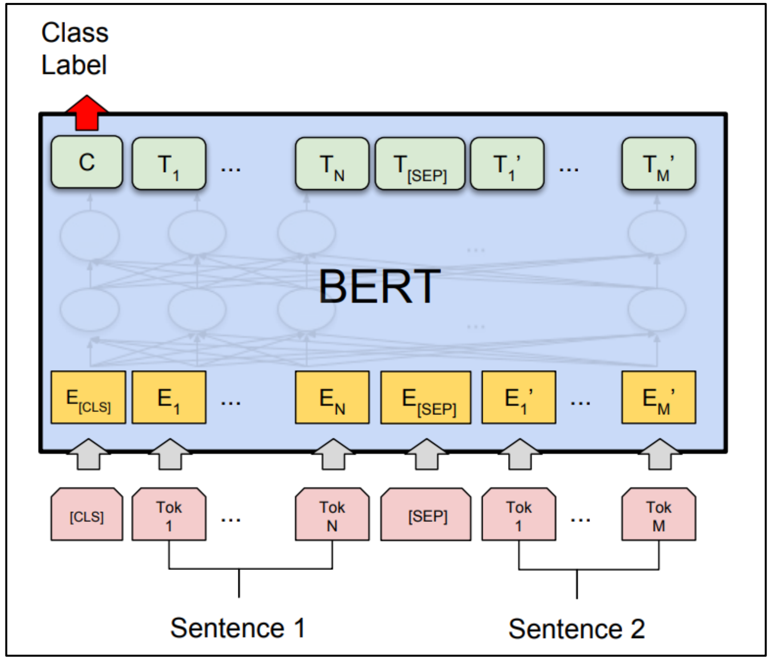
nodes	4(4x8 GPUs)		8(8x8 GPUs)		16(16x8 GPUs)		32(32x8 GPUs)	
topology	acc.	time	acc.	time	acc.	time	acc.	time
P-SGD	76.32	11.6	76.47	6.3	76.46	3.7	76.25	2.2
Ring	76.16	11.6	76.14	6.5	76.16	3.3	75.62	1.8
one-peer exp.	76.34	11.1	76.52	5.7	76.47	2.8	76.27	1.5

DSGD over ring has more efficient comm. than PSGD; **suffers from performance degradation**

DSGD over one-peer exp. graph is more comm.-efficient **without performance degradation**

[YYC+21]B. Ying, K. Yuan, Y. Chen, H. Hu, P. Pan, and W. Yin, ``Exponential Graph is Provably Efficient for Deep Training", NeurIPS 2021

Experiments in deep training (language modeling)



Model: BERT-Large (330M parameters)

Dataset: Wikipedia (2500M words) and

BookCorpus (800M words)

Hardware: 64 GPUs

Table. Comparison in loss and training time [CYZ+21]

Method	Final Loss	Wall-clock Time (hrs)
P-SGD	1.75	59.02
D-SGD	1.77	30.4

[CYZ+21] Y. Chen, K. Yuan, Y. Zhang, P. Pan, Y. Xu, and W. Yin, ``Accelerating Gossip SGD with Periodic Global Averaging", ICML 2021

A brief summary

- Exponential graphs are both sparse and effective. They are nearly best up to logarithm terms
- One-peer exponential graph is even sparser without hurting effectiveness

Topology	Per-iter. Comm.	Trans. Iters. (iid scenario)
Ring	$O(1)$	$O(n^7)$
2D-Grid	$O(1)$	$\tilde{O}(n^5)$
2D-Torus	$O(1)$	$O(n^5)$
$\frac{1}{2}$-RandGraph	$O(n)$	$O(n^3)$
Static Exp	$\tilde{O}(1)$	$\tilde{O}(n^3)$
One-Peer Expo	$O(1)$	$\tilde{O}(n^3)$

However …

- Periodic exact average for one-peer exp. **only holds** when network size **n is a power of 2**
- Not known when one-peer exp. performs well when n is **not** a power of 2
- Not known whether the transient iteration $O(n^3 \log_2^2(n))$ can be further improved to $O(n^3)$

Can we develop topologies that

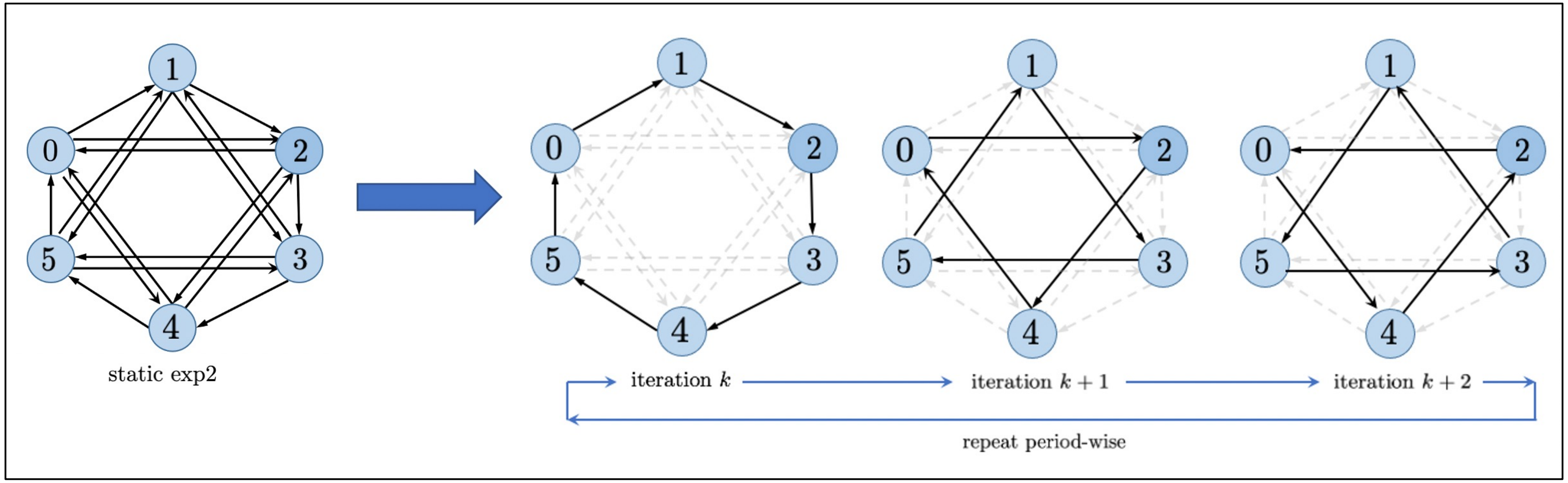
- have $O(1)$ per-iteration communication cost;
- enable DSGD to converge with $O(n^3)$ transient iteration complexity;
- and are valid for any network size n?

PART 03

EquiTopo graphs are new state-of-the-art

Why does exponential graph suffer log(n) deterioration?

- Exponential graphs are still not well-connected



- For example, node 0 never sends messages to nodes 3 and 5
- We need to develop topologies that every pair of nodes is connected in positive probability

Basis weight matrix and graph

Definition Given a graph of size n, we introduce a set of doubly stochastic *basis matrices* $\{A^{(u,n)}\}_{u=1}^{n-1}$, where $A^{(u,n)} = [a_{ij}^{(u,n)}] \in \mathbb{R}^{n\times n}$ with

$$a_{ij}^{(u,n)} = \begin{cases} \frac{n-1}{n}, & \text{if } i = (j+u) \mod n, \\ \frac{1}{n}, & \text{if } i = j, \\ 0, & \text{otherwise.} \end{cases}$$

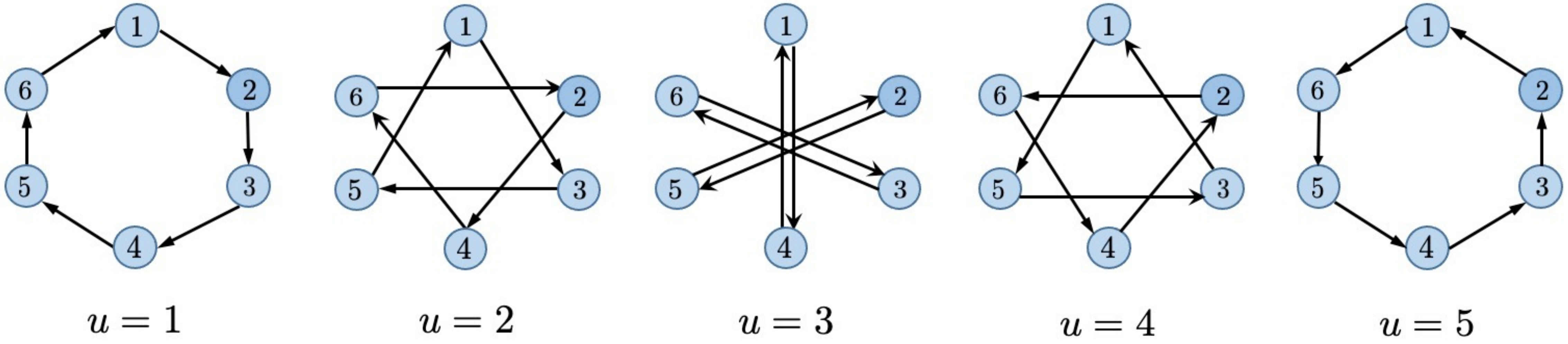
Their associated graphs $\{\mathcal{G}(A^{(u,n)})\}_{u=1}^{n-1}$ are called *basis graphs*.

Z. Song*, W. Li*, K. Jin*, L. Shi, M. Yan, W. Yin, and K. Yuan "Communication-efficient topologies for decentralized learning with O(1) consensus rate", NeurIPS 2022

Basis weight matrix and graph: illustration

PEKING UNIVERSITY 北京大学

The set of basis graphs $\{\mathcal{G}(A^{(u)})\}_{u=1}^{5}$ for n = 6



- Each basis graph $\mathcal{G}(A^{(u)})$ is an **one-peer** graph; O(1) per-iteration communication overhead
- Each edge in a basis graph has the same **label difference**

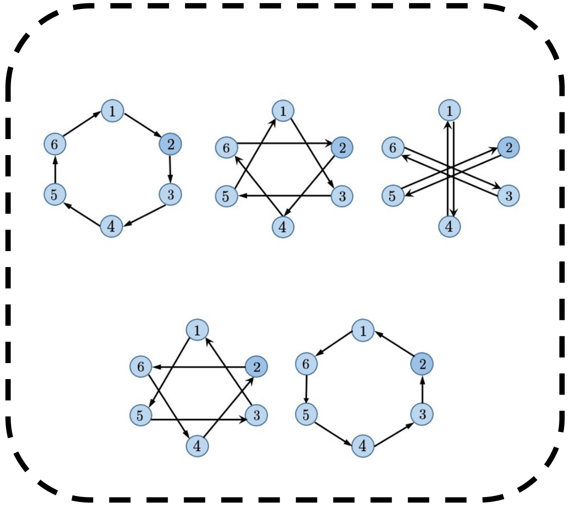
OP-EquiDyn: One-peer dynamic graph with equivalent label differences

Generate EquiDyn realization $W^{(k)}$

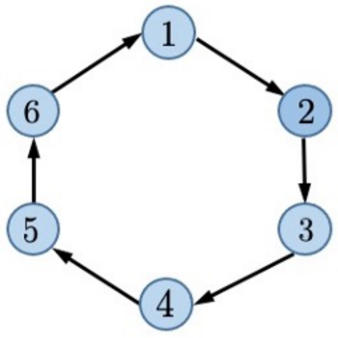
Pick v_k from uniform distribution over the basis index set $[n-1]$

Produce basis matrix $A^{(v_k)}$ according to the definition

Generate weight matrix $W^{(k)} = (1-\eta)I + \eta A^{(v_k)}$



Sample at k



$$W = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

OP-EquiDyn: One-peer dynamic graph with equivalent label differences

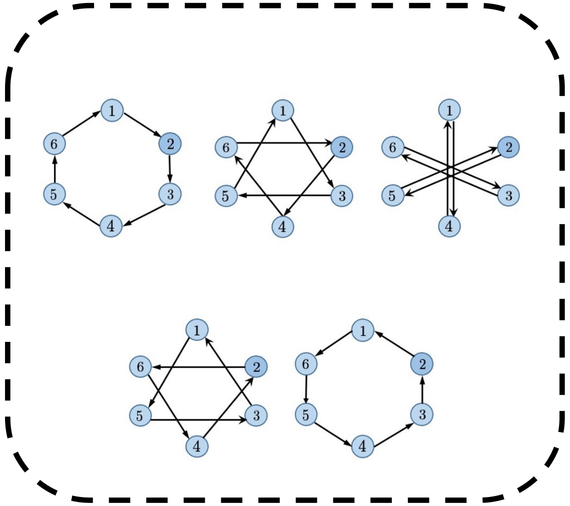
北京大学 PEKING UNIVERSITY

Generate EquiDyn realization $W^{(k)}$

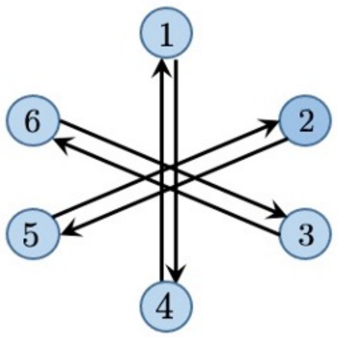
Pick v_k from uniform distribution over the basis index set $[n-1]$

Produce basis matrix $A^{(v_k)}$ according to the definition

Generate weight matrix $W^{(k)} = (1-\eta)I + \eta A^{(v_k)}$



Sample at k+1



$$W = \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

OP-EquiDyn: One-peer dynamic graph with equivalent label differences

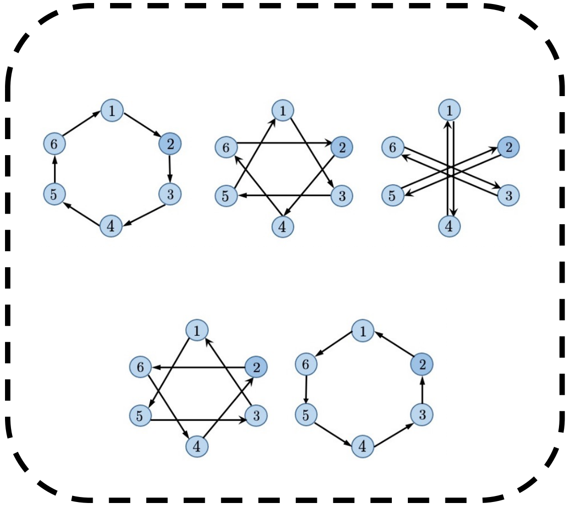
北京大学 PEKING UNIVERSITY

Generate EquiDyn realization $W^{(k)}$

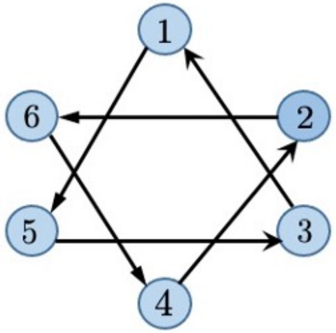
Pick v_k from uniform distribution over the basis index set $[n-1]$

Produce basis matrix $A^{(v_k)}$ according to the definition

Generate weight matrix $W^{(k)} = (1-\eta)I + \eta A^{(v_k)}$



Sample at k+2



$$W = \begin{bmatrix} \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

OP-EquiDyn v.s. OP-Exponential

OP-Exp

Sampled in a **cyclic** manner

Nodes with **exponential** label differences can be connected

OP-EquiDyn

Sampled in a **random** manner

Nodes with **any** label differences can be connected

OP-EquiDyn has a network-size-independent spectral gap

北京大學 PEKING UNIVERSITY

Theorem. Let the one-peer directed weight matrix $W^{(k)}$ be generated by the above EquiDyn algorithm. If we let $\eta = 1/2$, it then holds that

$$\rho = \mathbb{E}\|W^{(k)} - \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^T\|_2 \le \frac{\sqrt{2}}{2}$$

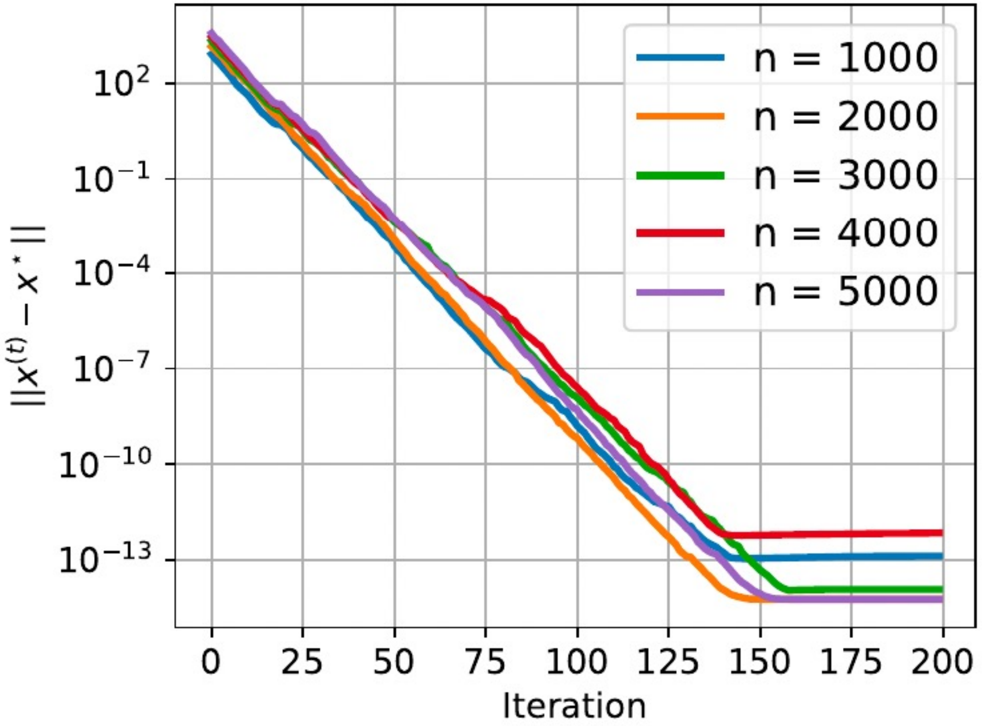
- Such spectral gap is **independent of network size**, and holds for **any size n**
- Recall that DSGD has transient iteration complexity $O(n^3(1-\rho)^{-2})$
- Substituting $\rho = \sqrt{2}/2$, DSGD over OP-EquiDyn has tran. iters. $O(n^3)$

OP-EquiDyn has a network-size-independent spectral gap

- We examine $\|\frac{1}{n}\mathbb{1}\mathbb{1}^T x - \prod_{k=0}^{T} W^{(k)} x\|$ for a vector x

- OP-EquiDyn has a network-size-independent rate

- While network size increases, consensus rate remains almost unchanged



OP-EquiDyn achieves new SOTA results

DSGD with different network topology

Topology	Per-iter. Comm.	Trans. Iters. (iid scenario)
Ring	$O(1)$	$O(n^7)$
2D-Grid	$O(1)$	$\tilde{O}(n^5)$
2D-Torus	$O(1)$	$O(n^5)$
$\frac{1}{2}$-RandGraph	$O(n)$	$O(n^3)$
Static Exp	$\tilde{O}(1)$	$\tilde{O}(n^3)$
One-Peer Expo	$O(1)$	$\tilde{O}(n^3)$
O.-P. EquiDyn	$O(1)$	$O(n^3)$

- OP-EquiDyn achieves O(1) comm., $O(n^3)$ transient iteration complexity, and holds for any size n
- Since DSGD has a transient complexity as $O(n^3(1-\rho)^{-2})$, the order $O(n^3)$ cannot be improved

OP-EquiDyn can also accelerate other decentralized methods

北京大学 PEKING UNIVERSITY

Gradient tracking

$$\boldsymbol{x}_i^{(t+1)} = \sum_{j=1}^n w_{ij}^{(t)} \left(\boldsymbol{x}_j^{(t)} - \gamma \boldsymbol{y}_j^{(t)}\right);$$
$$\boldsymbol{y}_i^{(t+1)} = \sum_{j=1}^n w_{ij}^{(t)} \boldsymbol{y}_j^{(t)} + \boldsymbol{g}_i^{(t+1)} - \boldsymbol{g}_i^{(t)}, \quad \boldsymbol{y}_i^{(0)} = \boldsymbol{g}_i^{(0)}.$$

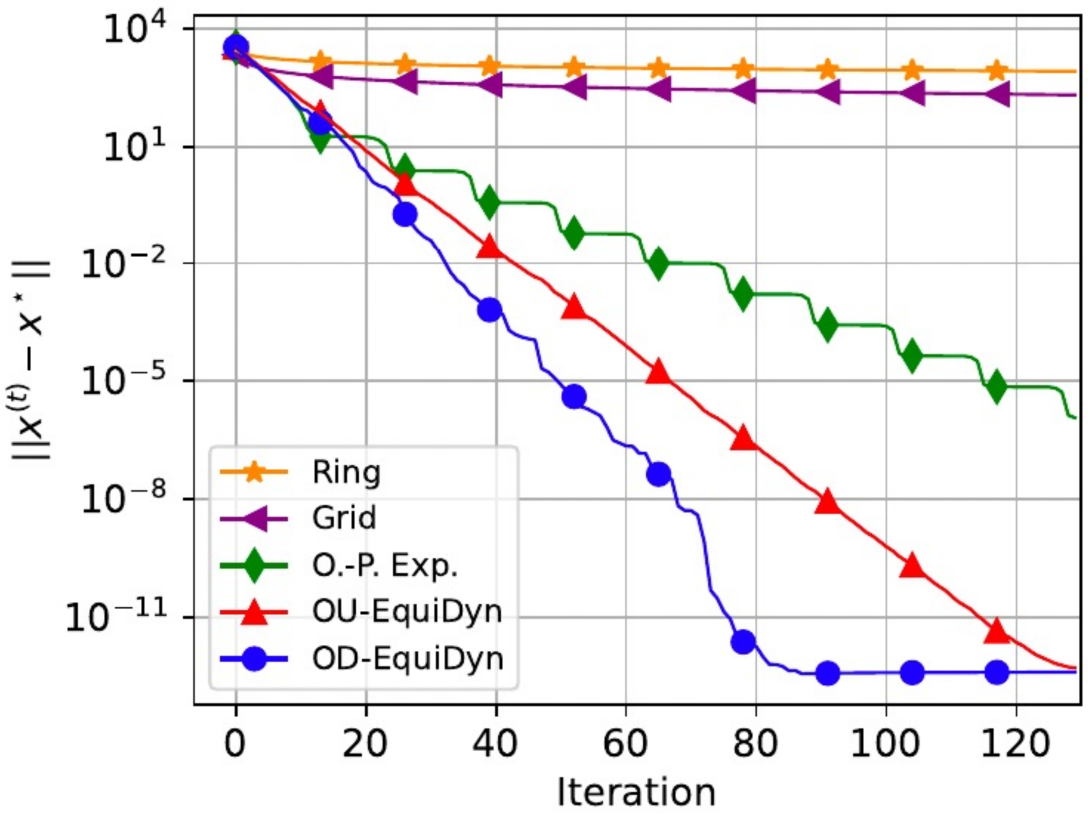
Topology	Per-iter Comm.	Convergence Rate	Trans. Iters.
Ring	$\Theta(1)$	$\mathcal{O}\left(\frac{\sigma}{\sqrt{nT}} + \frac{n^2\sigma^{\frac{2}{3}}}{T^{\frac{2}{3}}} + \frac{n^4}{T}\right)$	$\mathcal{O}(n^{15})$
Torus	$\Theta(1)$	$\mathcal{O}\left(\frac{\sigma}{\sqrt{nT}} + \frac{n\sigma^{\frac{2}{3}}}{T^{\frac{2}{3}}} + \frac{n^2}{T}\right)$	$\mathcal{O}(n^9)$
Static Exp.	$\Theta(\ln(n))$	$\mathcal{O}\left(\frac{\sigma}{\sqrt{nT}} + \frac{\ln(n)\sigma^{\frac{2}{3}}}{T^{\frac{2}{3}}} + \frac{\ln^2(n)}{T}\right)$	$\mathcal{O}(n^3 \ln^6(n))$
O.-P. Exp.	1	$\mathcal{O}\left(\frac{\sigma}{\sqrt{nT}} + \frac{\ln(n)\sigma^{\frac{2}{3}}}{T^{\frac{2}{3}}} + \frac{\ln^2(n)}{T}\right)$	$\mathcal{O}(n^3 \ln^6(n))$
OD (OU)-EquiDyn	1	$\mathcal{O}\left(\frac{\sigma}{\sqrt{nT}} + \left(\frac{\sigma}{T}\right)^{\frac{2}{3}} + \frac{1}{T}\right)$	$\mathcal{O}(n^3)$

Experiments: compare with other topologies

- We compare consensus rate (i.e., spectral gap) between various topologies

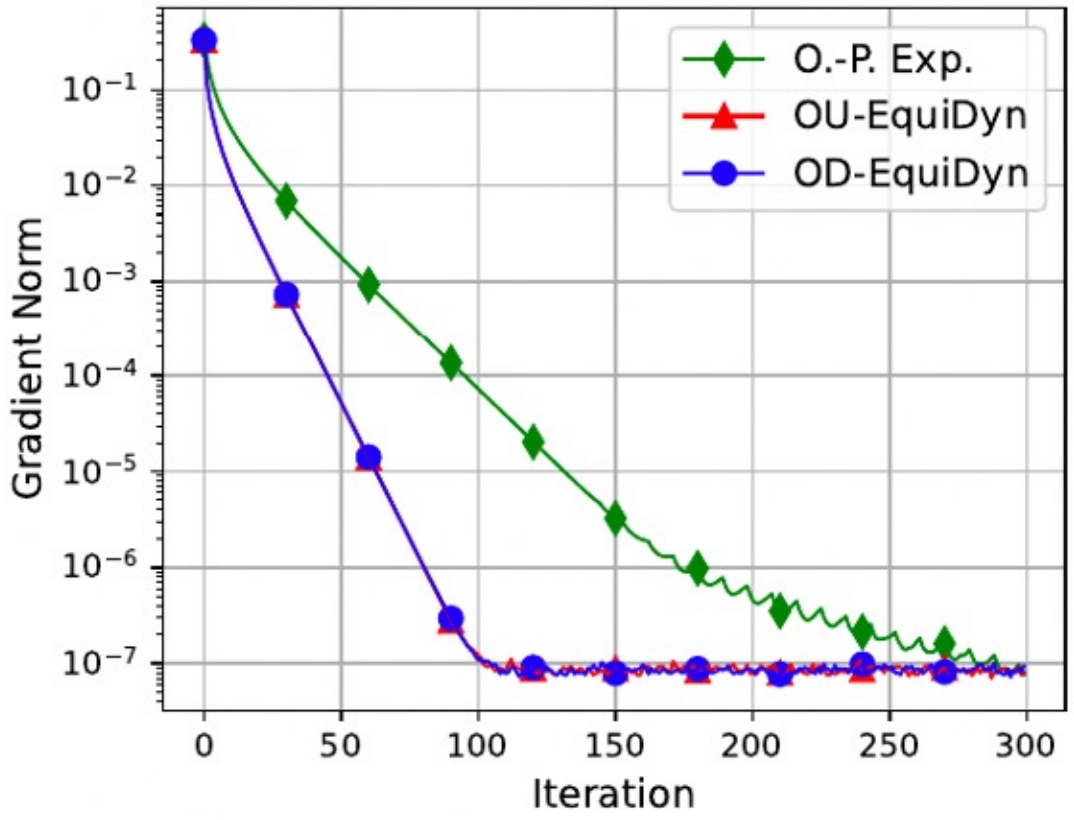
- Network size is 4900

- EquiDyn converges the fastest



Experiments: gradient tracking with different topologies

- We use GT to solve logistic regression with non-convex regularizes
- Network size is 300
- GT with EquiDyn converges faster than OP-Exp



Experiments: deep learning experiments

- EquiTopo graph has many variants, i.e., OU-EquiDyn supports undirected graphs
- EquiTopo graph outperforms other common topologies with 17 GPUs

Topology	MNIST Acc.	CIFAR-10 Acc.
Centralized SGD	98.34	91.76
Ring	98.32	91.25
Static Exp.	98.31	91.48
O.-P. Exp.	98.17	90.86
D-EquiStatic	98.29	**92.01**
U-EquiStatic	98.26	91.74
OD-EquiDyn	**98.39**	91.44
OU-EquiDyn	98.12	91.56

Summary

Can we develop topologies that

- have $O(1)$ per-iteration communication cost;
- enable DSGD to converge with $O(n^3)$ transient iteration complexity;
- and are valid for any network size n?

One-peer EquiDyn is the answer!

北京大学 PEKING UNIVERSITY

PART 04

BlueFog: An open-source and high-performance python library

BlueFog

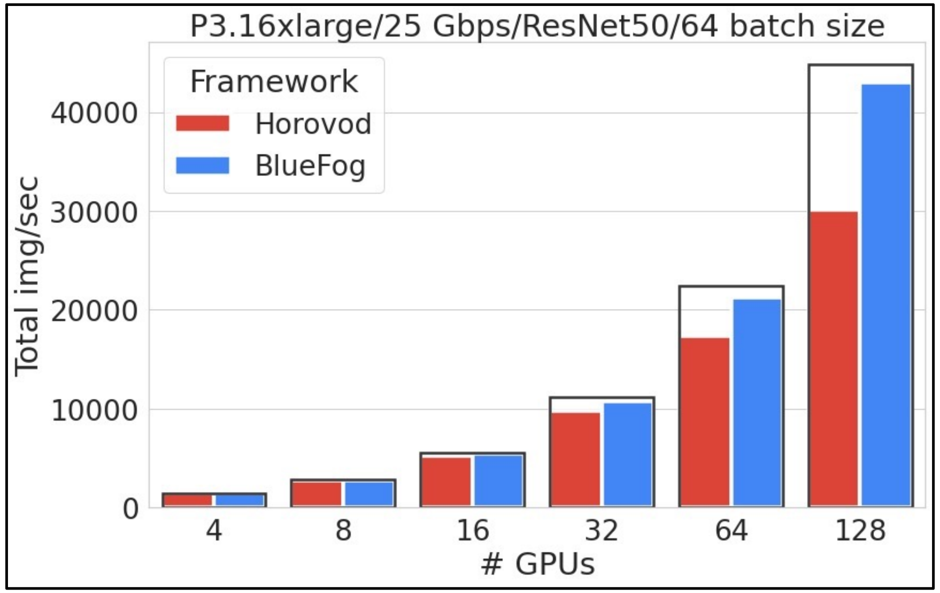
https://github.com/Bluefog-Lib/bluefog

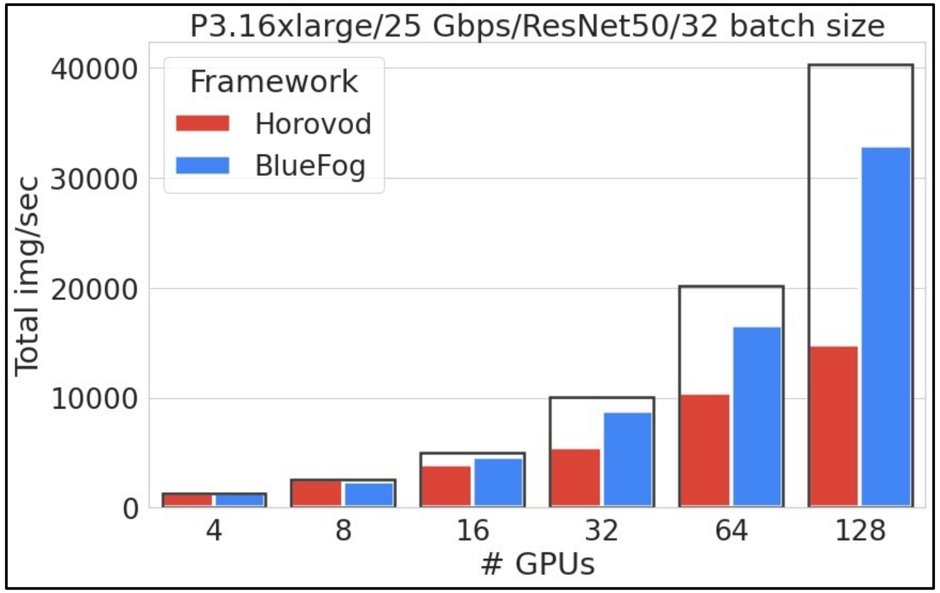
BlueFog

- An open-source library to support decentralized communication in optimization and deep learning
- High-performance
- Easy-to-use

High-performance

- BlueFog has larger throughput than Horovod (the SOTA DL system implementing PSGD) [YYH+21]





- All our research progresses are involved in BlueFog

[YYH+21] B. Ying, K. Yuan, H. Hu, Y. Chen, and W. Yin, ``BlueFog: Make Decentralized Algorithms Practical for Optimization and machine learning", arXiv:2111.04287 [GitHub site: github.com/Bluefog-Lib/bluefog]

Easy-to-use

- Writing codes for decentralized methods is as easy as writing equations

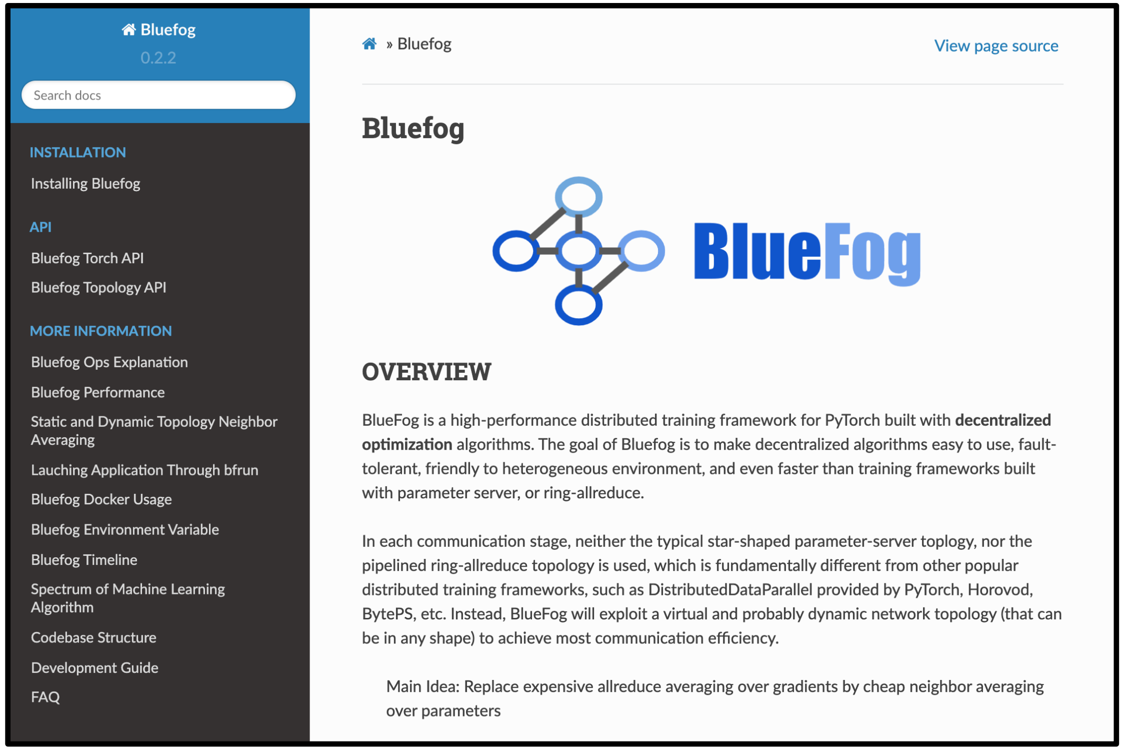
Decentralized least-square algorithms

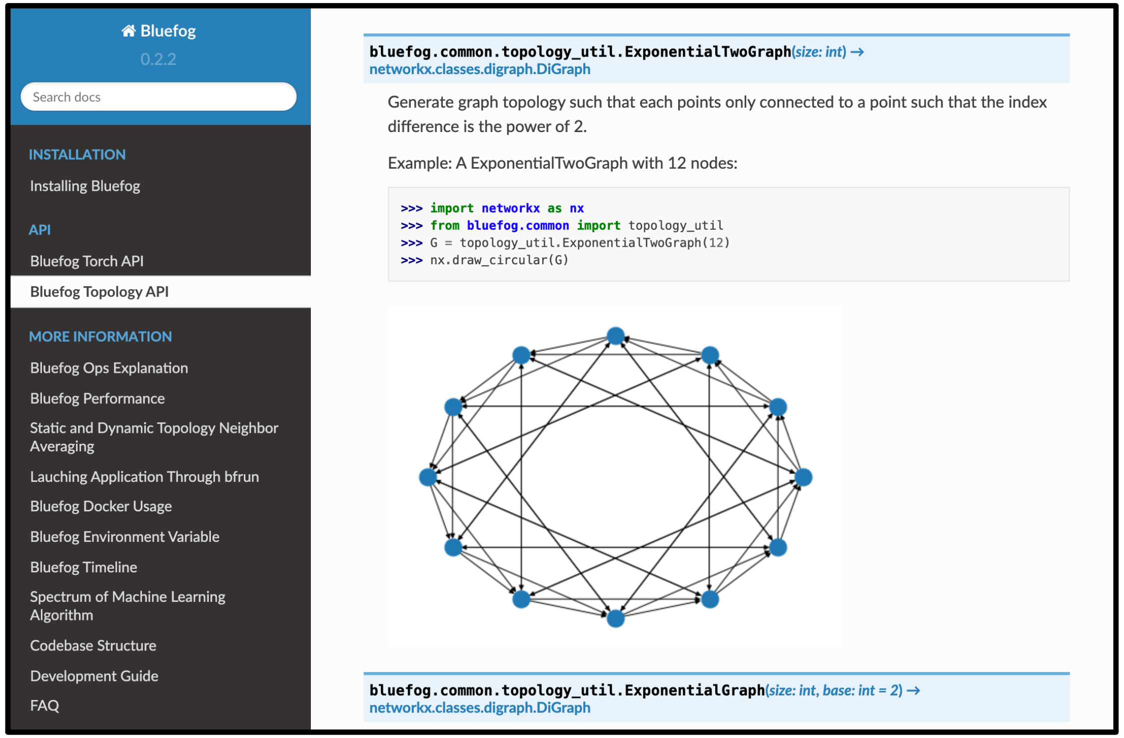
$$y_i^{(k)} = x_i^{(k)} - \gamma A_i^T (A_i x_i^{(k)} - b_i)$$
$$x_i^{(k+1)} = \sum_{j \in \mathcal{N}_i} w_{ij} y_j^{(k)}$$

```python
import bluefog.torch as bf
bf.init()   # Initialize the BlueFog

# Set topology as static exponential graph.
G = bf.ExponentialTwoGraph(bf.size())
bf.set_topology(G)

# DGD implementation
for ite in range(maxite):
    grad_local = A.t().mm(A.mm(x) - b)   # compute local grad
    y = x - gamma * grad_local           # local update
    x = bf.neighbor_allreduce(y)         # partial averaging
```

Easy-to-use

Abundant documents





Easy-to-use

北京大学
PEKING UNIVERSITY

Detailed tutorials

Contents

2.1.3 Initialize BlueFog and test it

All contents in this section are displayed in Jupyter notebook, and all experimental examples are written with BlueFog and iParallel. Readers not familiar with how to run BlueFog in ipython notebook environment is encouraged to read Sec. [HelloWorld section] first. In the following codes, we will initialize BlueFog and test whether it works normally.

The output of `rc.ids` should be a list from 0 to the number of processes minus one. The number of processes is the one you set in the `ibfrun start -np {X}`.

In [1]:
```
import ipyparallel as ipp

rc = ipp.Client(profile="bluefog")
rc.ids
```

Let each agent import necessary modules and then initialize BlueFog. You should be able to see the printed information like:

> [stdout:0] Hello, I am 1 among 4 processes
>
> ...

In [2]:
```
%%px
import numpy as np
import bluefog.torch as bf
import torch
from bluefog.common import topology_util
import networkx as nx

bf.init()
print(f"Hello, I am {bf.rank()} among {bf.size()} processes")
```

Push seed to each agent so that the simulation can be reproduced.

In [3]:
```
dview = rc[:]  # A DirectView of all engines
dview.block = True

# Push the data into all workers
#   `dview.push({'seed': 2021}, block=True)`
# Or equivalently
dview["seed"] = 2021
```

After running the following code, you should be able to see the printed information like

> [stdout:0] I received seed as value: 2021

Final summary

- Decentralized algorithms save remarkable communication compared to centralized ones
- Sparse and effective topologies make decentralized optimization practical for deep training
- We propose static exponential, one-peer exponential, and one-peer EquiDyn and justify their superiority with strong theoretical and experimental evidences
- We introduced BlueFog to facilitate research and implementation of decentralized methods

PEKING UNIVERSITY

Thank you!

Kun Yuan homepage: https://kunyuan827.github.io/

BlueFog homepage: https://github.com/Bluefog-Lib/bluefog